GDG Astrakhan

DevFest

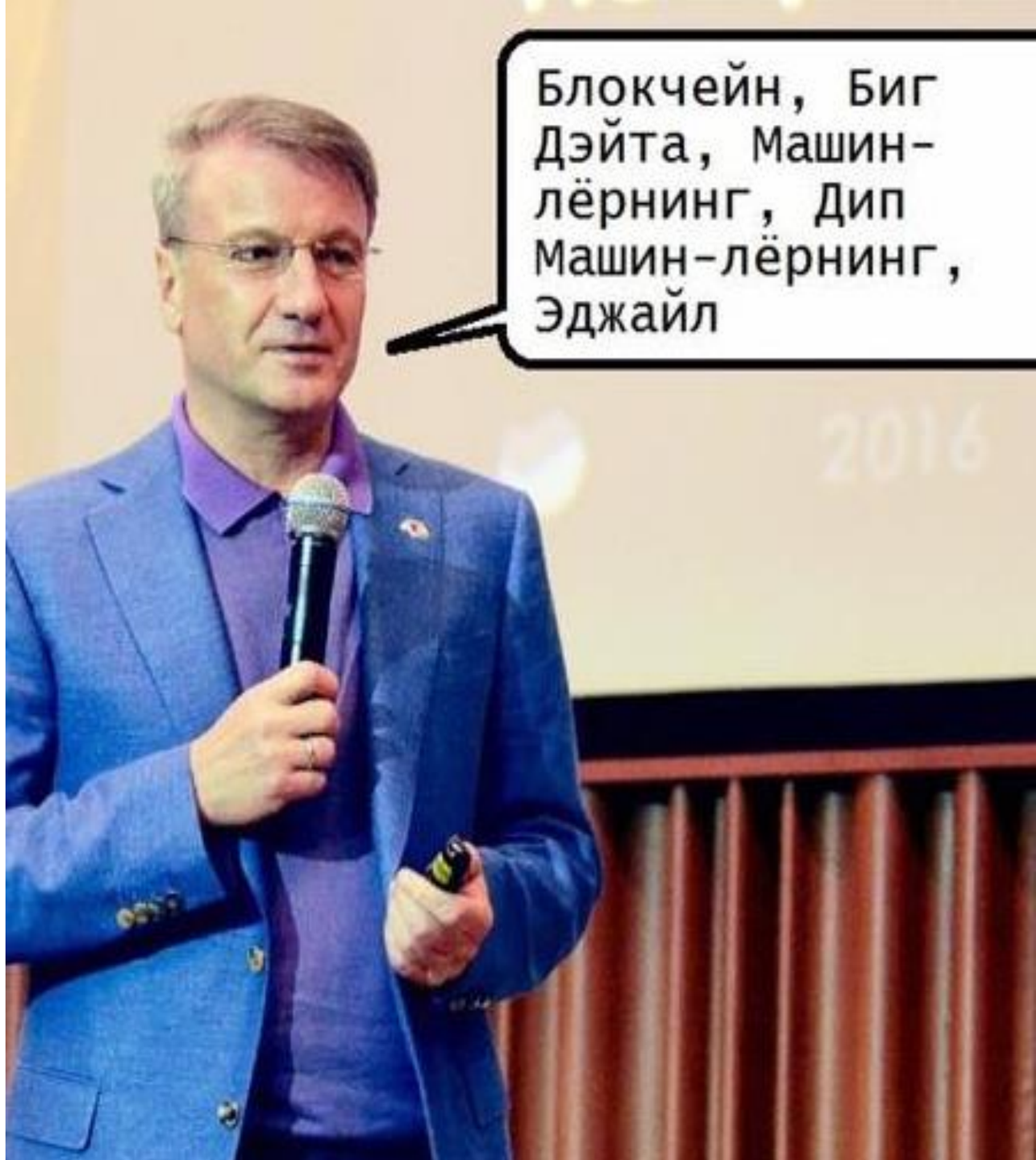
Искусственный интеллект. Антихайп версия.

Александр Морозов

Зачем доклад

- меньше лапши на ушах
- больше адекватных идей

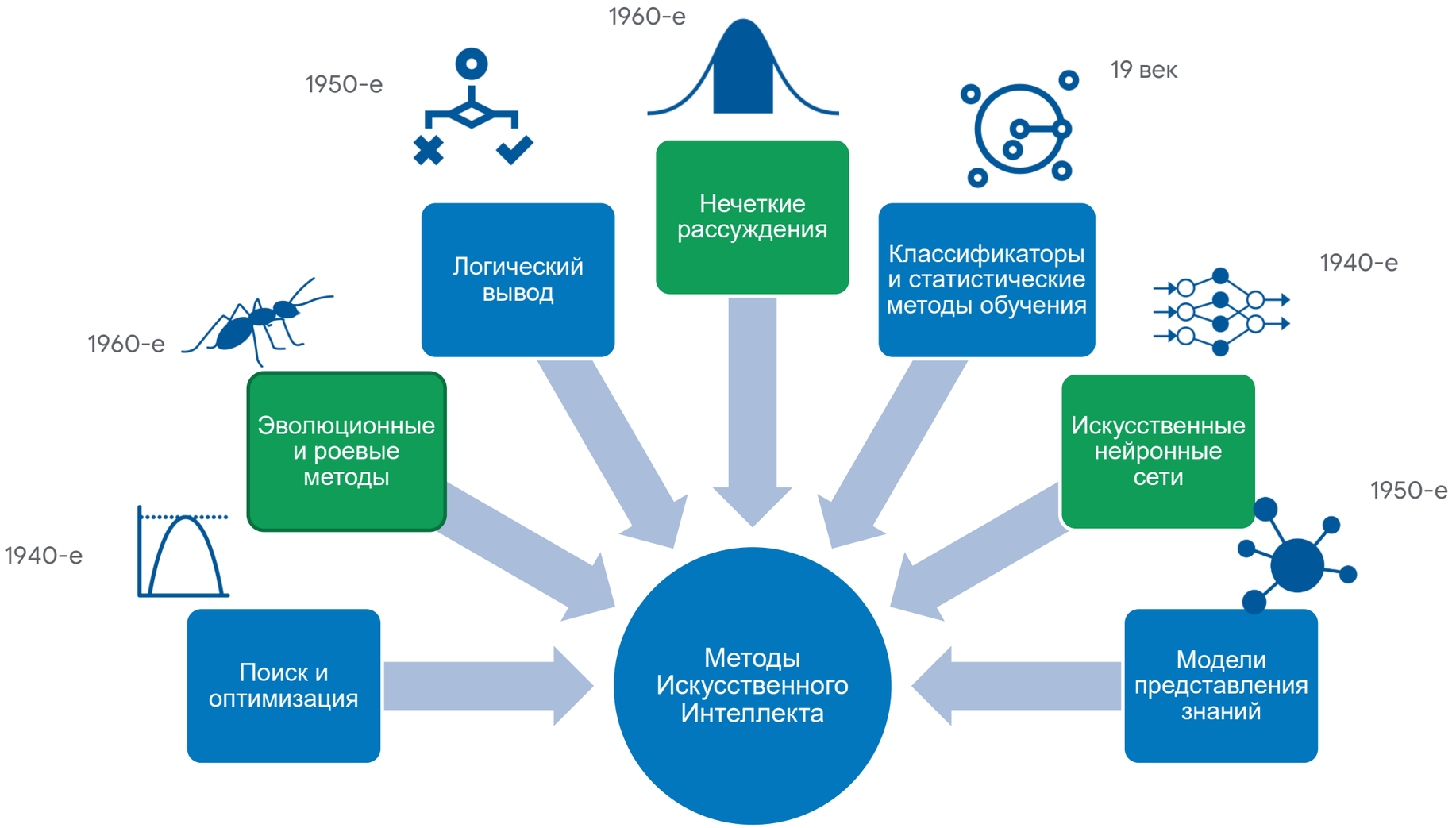
- меньше магии
- больше реальных компетенций



Понятие об искусственном интеллекте

МЫ ВСЕ УМРЕМ…

Искусственный интеллект = современная вычислительная математика

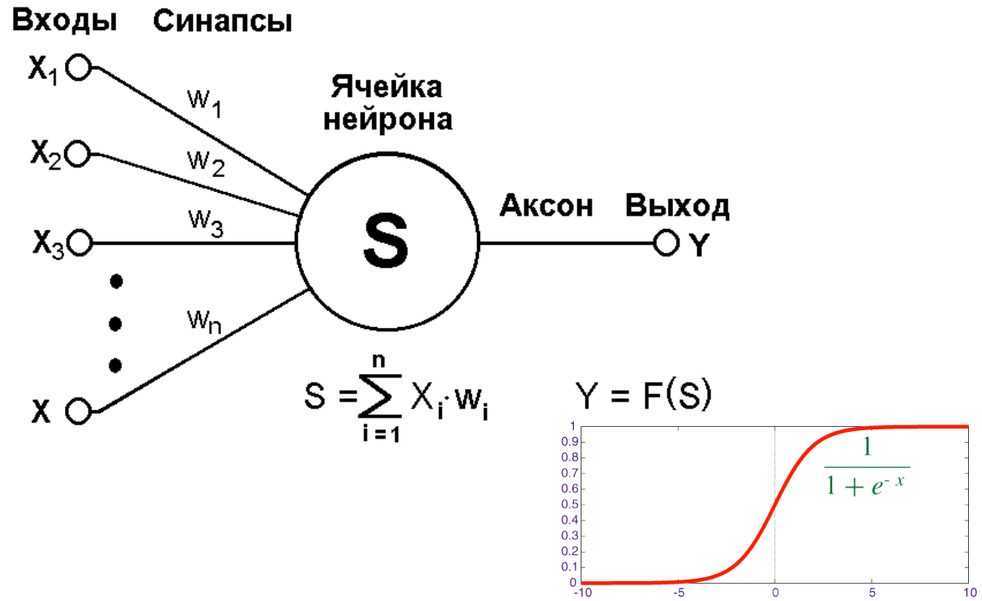


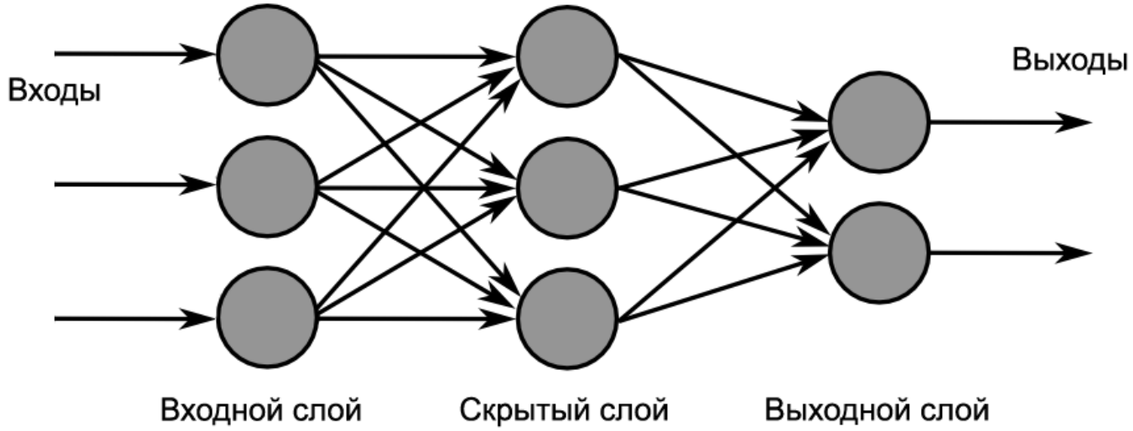

Нейронные сети

Нейронные сети

- придуманы в попытках имитировать мозг человека
- решают творческие задачи
- скоро будут использоваться везде
- никто не знает как работают

Как работает нейронная сеть





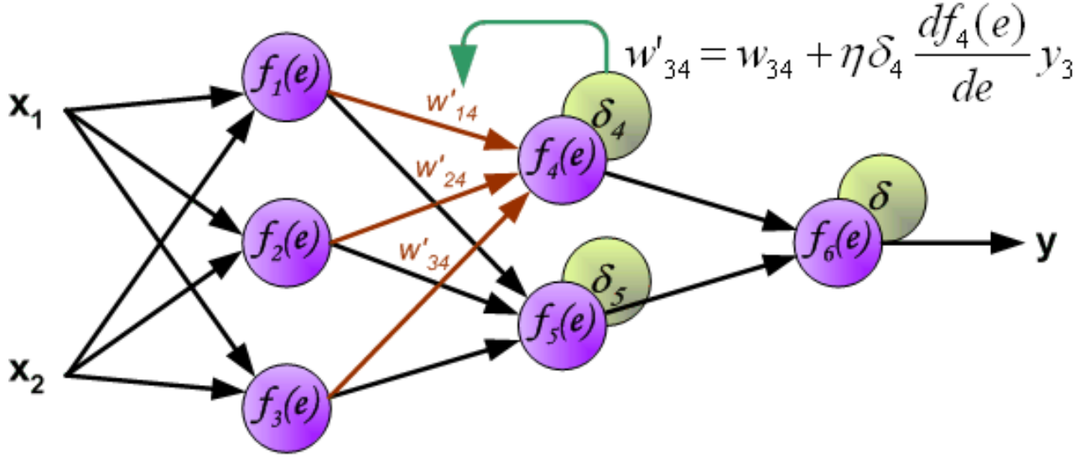
Чтобы понять как работает нейронная сеть – нужно освоить учебник биологии и математики за 9 класс

Как учится нейронная сеть

$$w'_{14} = w_{14} + \eta\delta_4 \frac{df_4(e)}{de} y_1$$

$$w'_{24} = w_{24} + \eta\delta_4 \frac{df_4(e)}{de} y_2$$

$$w'_{34} = w_{34} + \eta\delta_4 \frac{df_4(e)}{de} y_3$$



Если нейронная сеть выдала не то что вы хотели – покажите ей, что вы ожидали и попросите больше так не ошибаться.

Чтобы понять как учится нейронная сеть – нужно знать производные и метод наименьших квадратов. Это первый курс, первый семестр если вы помните.

Как сделать нейронную сеть

```
INeuralNetwork network = NetworkManager.NewSequential(TensorInfo.Image<Alpha8>(28, 28),
NetworkLayers.Convolutional((5, 5), 20, ActivationType.Identity),
NetworkLayers.Pooling(ActivationType.LeakyReLU),
NetworkLayers.Convolutional((3, 3), 40, ActivationType.Identity),
NetworkLayers.Pooling(ActivationType.LeakyReLU),
NetworkLayers.FullyConnected(125, ActivationType.LeakyReLU),
NetworkLayers.FullyConnected(64, ActivationType.LeakyReLU),
NetworkLayers.Softmax(10));
```

```
TrainingSessionResult result = NetworkManager.TrainNetwork(
network, // The network instance to train
dataset, // The ITrainingDataset instance
TrainingAlgorithms.AdaDelta(), // The training algorithm to use
60, // The expected number of training epochs to run
0.5f, // Dropout probability
test, // Test dataset
token); // Cancellation token for the training
```

Что могут и не могут нейронные сети

Могут

- классифицировать
- кластеризовать
- апроксимировать
- прогнозировать

Не могут

- выявлять закономерности
- объяснять почему они выдали тот или иной ответ
- придумывать языки для общения между собой ☺

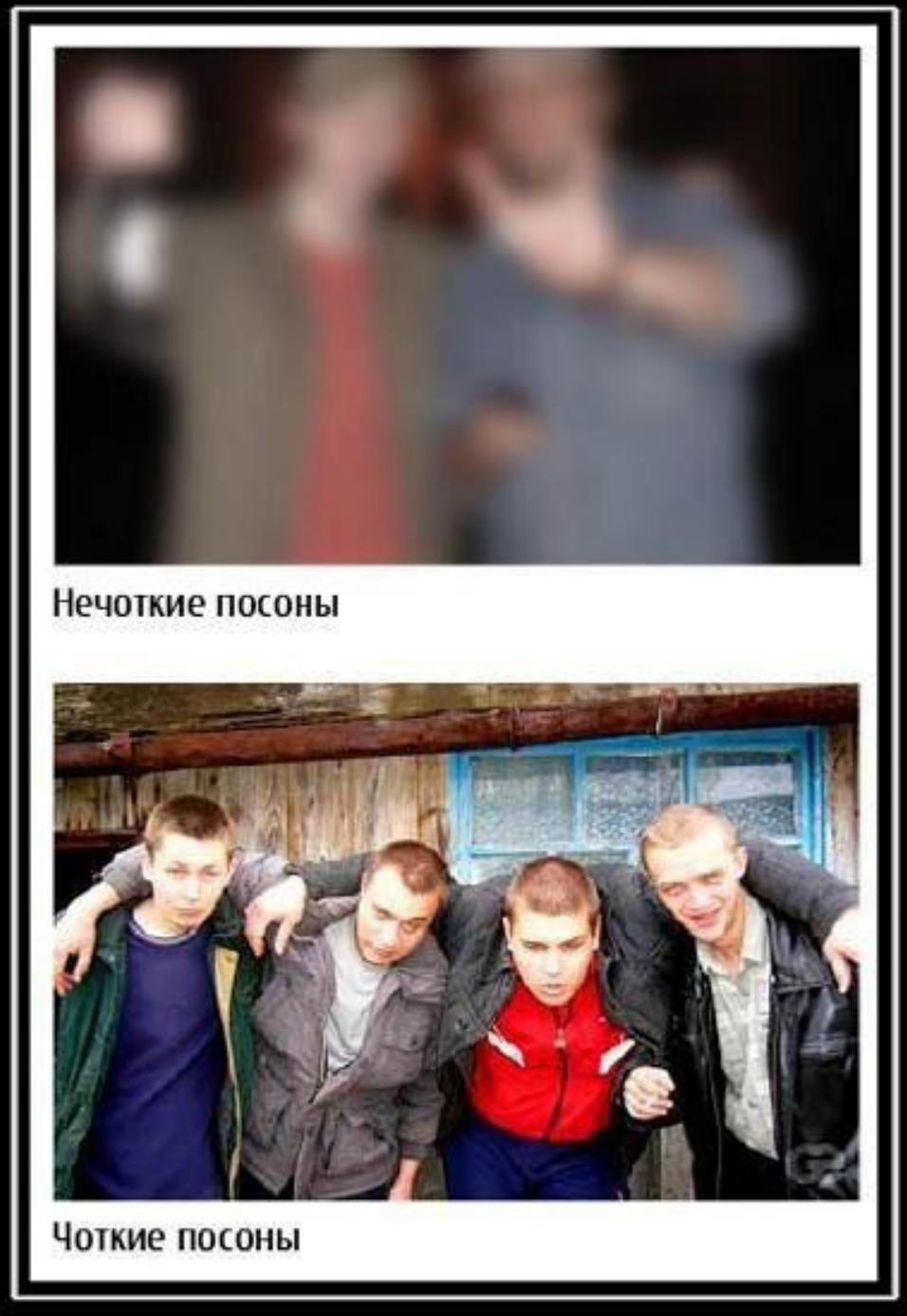
Реальные причины использования нейронных сетей

- это модно, стильно, молодежно
- нормально формализовать задачу и выяснить реальные зависимости факторов дорого или лень
- нет хороших данных, чтобы построить нормальную статистическую модель
- нужно для того, чтобы сравнить с другими методами
- нужно, потому что другие способы медленнее считают, а нужно в реальном времени

Нечеткая логика

Нечеткая логика

- придумана для того чтобы отличать больных от здоровых
- сложная альтернатива обычной статистике и теории вероятностей
- на самом деле нигде не используется (везде вместо них нейронные сети)



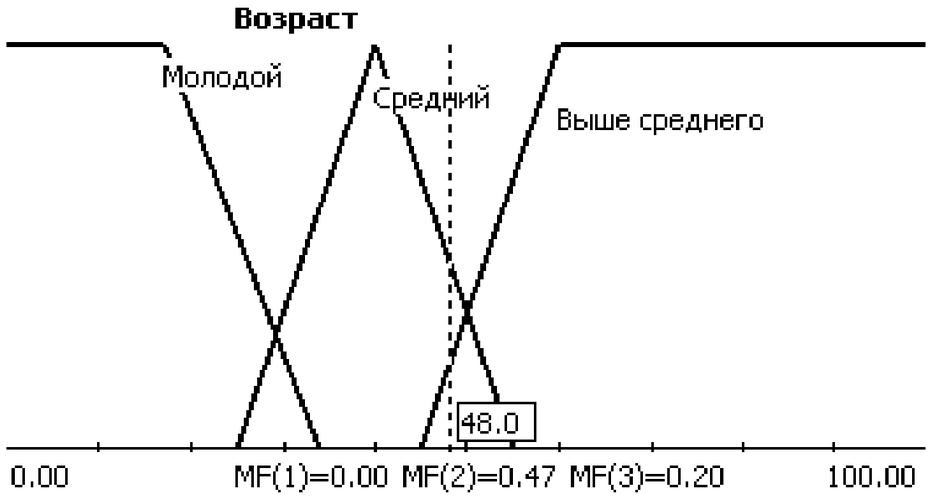
Как устроена нечетка логика

- Если Возраст Выше среднего и Температура Высокая То Госпитализировать

- Если Возраст Молодой То Лечение на дому

- Если Возраст Средний И Температура Повышенная То Не лечить
- ....

И = min (x1, x2)
ИЛИ = max(x1, x2)
НЕ = 1 – x1



Как реализовать нечеткие вычисления

```
var water = new LinguisticVariable("Water");
var cold = water.MembershipFunctions.AddTrapezoid("Cold", 0, 0, 20, 40);
var warm = water.MembershipFunctions.AddTriangle("Warm", 30, 50, 70);
var hot = water.MembershipFunctions.AddTrapezoid("Hot", 50, 80, 100, 100);

var power = new LinguisticVariable("Power");
var low = power.MembershipFunctions.AddTriangle("Low", 0, 25, 50);
var high = power.MembershipFunctions.AddTriangle("High", 25, 50, 75);

IFuzzyEngine fuzzyEngine = new FuzzyEngineFactory().Default();

var rule1 = Rule.If(water.Is(cold).Or(water.Is(warm))).Then(power.Is(high));
var rule2 = Rule.If(water.Is(hot)).Then(power.Is(low));
fuzzyEngine.Rules.Add(rule1, rule2);

var result = fuzzyEngine.Defuzzify(new { water = 60 });
```

Что может и не может нечеткая логика

Может

- учитывать погрешности
- описывать поведение в условиях неопределенности
- описывать правила на языке близком к естественному

Не может

- дать гарантию непротиворечивых результатов
- обеспечить понимание естественного языка машиной

Реальные причины использования нечеткой логики

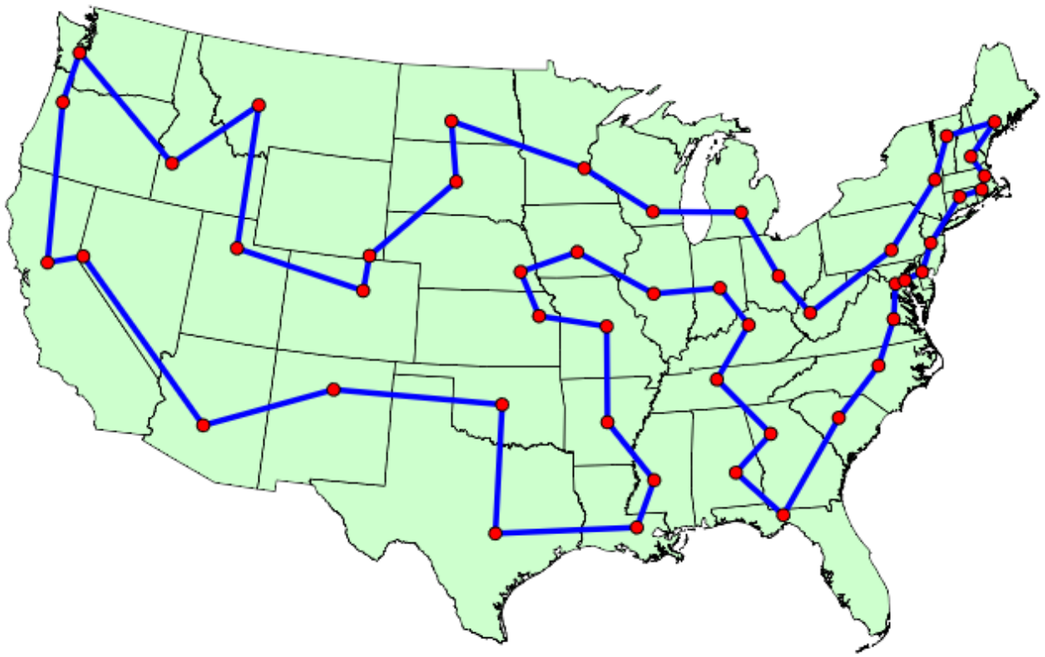
- лень использовать нормальную теорию вероятности (или ее забыли как страшный сон)
- нет статистически значимых данных
- эксперты из предметной области не знают математики, но хотят влиять на поведение системы

Эволюционные вычисления

Эволюционные вычисления

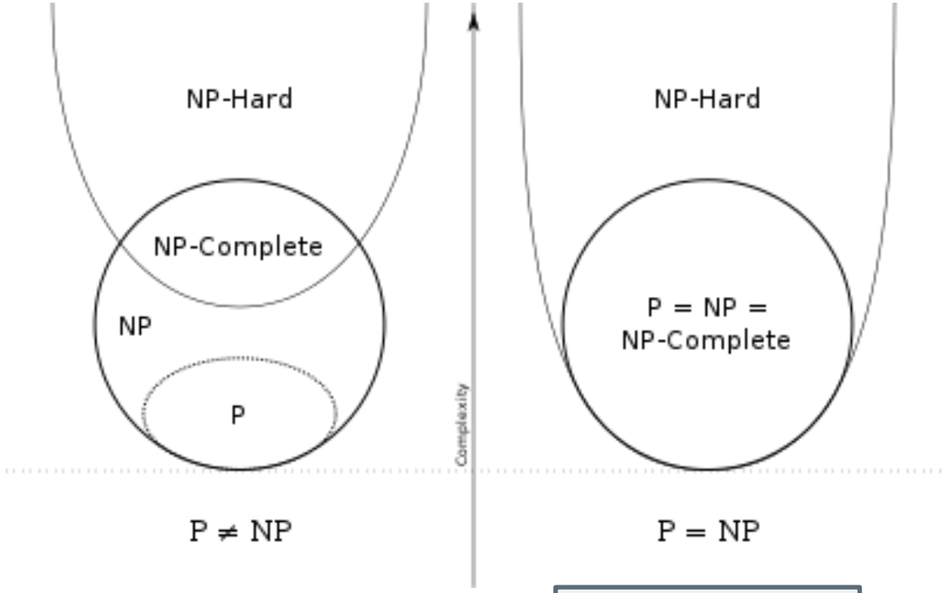
- это моделирование поведения природных систем
- требуют длительного времени
- позволяют программам работать все лучше со временем
- подразумевают изменение кода программы самой программой

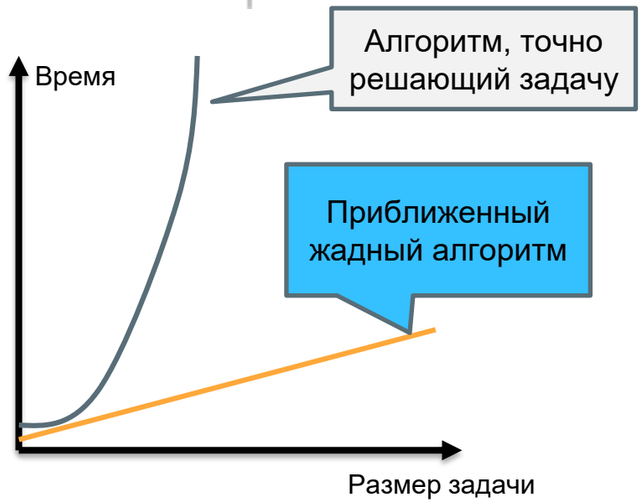
Когда нужны эволюционные вычисления



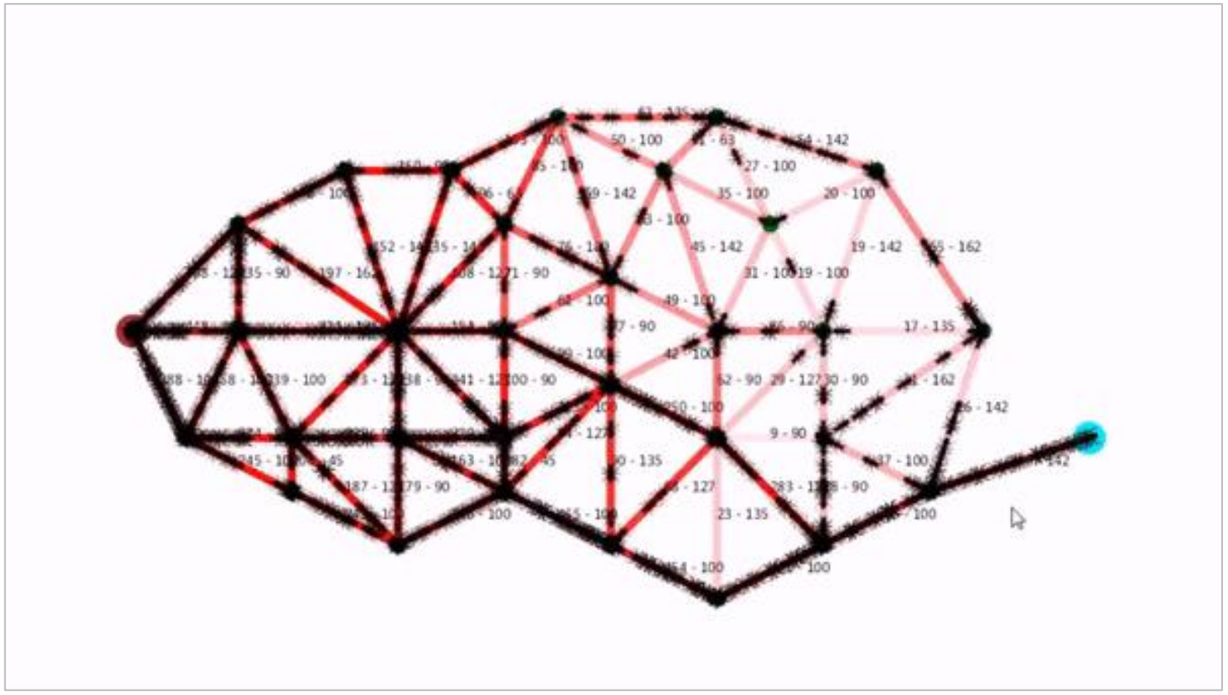
- Тут 42 города

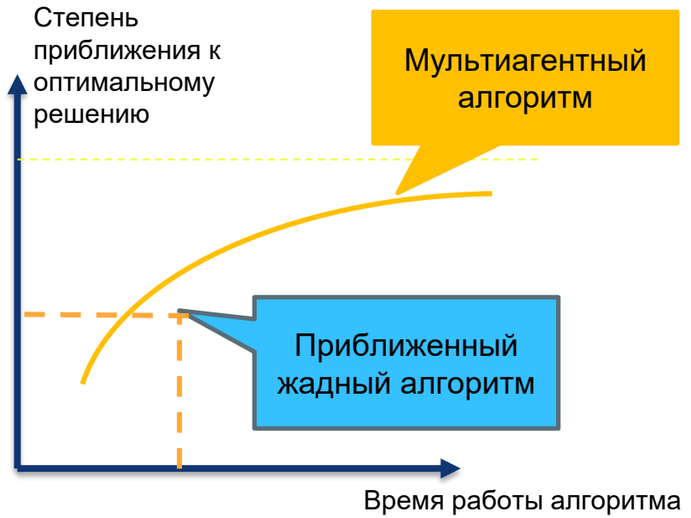
- Количество вариантов 1 405 006
  117 752 879 898 543 142 606 244
  511 569 936 384 000 000 000





Как работают эволюционные вычисления





Что могут и не могут эволюционные вычисления

Могут

- дать быстрый приближенный результат и постоянно его улучшать
- быть устойчивы к изменению входных данных
- масштабироваться горизонтально

Не могут

- точно решать задачи
- обходиться без простых жадных эвристик внутри
- масштабироваться линейно

Реальные причины использования эволюционных вычислений

- не нравятся формулы, нравятся алгоритмы,
- перебор не подходит, а жадные алгоритмы дают очень низкое приближение к оптимуму
- нравится чувствовать себя «творцом»
- реально нужно влиять на расчет в процессе его исполнения

Мораль

GDG Astrakhan

DevFest

Q&A

можно пообщаться лично
+7 917 170 72 63
telegram, whatsapp

agile, scrum, microservices, AI,
OOA/OOD/OOP, C#, 1C,
диссертации, ремонт квартир.

GDG Astrakhan

DevFest

Thank you!

Alexander Morozov, Adeptik
morozov@adeptik.com